GRADIENT IMAGE GENERATOR HARDWARE/SOFTWARE CO-DESIGN

ABBAS HAGHI

UNIVERSITI TEKNOLOGI MALAYSIA

GRADIENT IMAGE GENERATOR HARDWARE/SOFTWARE CO-DESIGN

ABBAS HAGHI

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Engineering (Electrical - Computer and Microelectronic System)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

JULY 2012

To my parents for their love, endless support and encouragement.

ACKNOWLEDGEMENT

First and foremost, I would like to thank to my supervisor of this project, Dr. Muhammad Nadzir Marsono for his valuable guidance and advice. He inspired me greatly to work in this project. His willingness to motivate me contributed tremendously to my project.

Furthermore, I wish to express my gratitude towards my family especially my parents. There is no word I could write which is adequate to depict how much they mean to me. The reason that has been keeping me going on through these times.

Lastly, I must to thank to my friends and coursemates for their kind encouragement and constant support throughout the project.

Skudai, Johor - Malaysia
Abbas Haghi

ABSTRACT

This project proposes a software and hardware architecture for computing image gradients in order to reduce the input image size. The only way to transfer data in real time using lower speed wireless communication systems is to reduce the frame size; if a 24bit image is binarized the size will be reduced 24 times. In this project the Canny algorithm is analyzed and written in Matlab and C programming language for NiosII CPU. Then it is implemented in a Field Programmable Gate Array (FPGA) hardware and the timing result for every step is measured. Based on these timing results, a final co-design is proposed. The output image after processing is a binary image that is at least 24 times smaller than the original image. For a sample 98×183, 24bit image and a working frequency equal to 50MHz, total logic elements for final co-design increased about 4 times of software design, but execution time in co-design architecture is 19 times faster than software. The hardware implementation in this paper is done on Altera CycloneII FPGA board.

ABSTRAK

Projek ini mereka bentuk perisian dan perkakasan bagi pengiraan kecerunan imej untuk tujuan pengurangan saiz imej. Penghantaran data imej secara "real time" melalui komunikasi wayarles hanya dapat dilaksanakan sekiranya saiz frame imej dikurangkan. Contohnya, saiz imej 24 bit yang dibinarikan akan berkurangan sebanyak 24 kali. Projek ini menganalisis algoritma Canny dan menterjemahkannya kepada kod Matlab dan seterusnya pengatucaraan C untuk dilaksankan oleh pemproses NiosII. Rekabentuk perkakasan kemudiannya dilaksanakan menggunakan Field Programmable Gate Array (FPGA) dan analisa masa untuk setiap langkah diukur. Berdasarkan keputusan daripada analisa masa, co-design peringkat akhir dicadangkan. Imej ouput daripada pemprosesan merupakan imej binari yang sekurang-kurangnya 24 kali lebih kecil daripada imej asal. Untuk sampel imej 24 bit dengan resolusi 98×183 dan 50MHz frekuensi operasi , jumlah elemen logik bagi rekabentuk akhir meningkat kepada 4 kali ganda berbanding dengan rekabentuk yang menggunakan perisian sahaja. Walau bagaimanapun, masa pemprosesan rekabentuk akhir adalah 19 kali lebih cepat berbanding menggunakan perisian. Rekabentuk perkakasan akhir untuk projek ini dilaksanakan dengan menggunankan papan Altera CycloneII FPGA.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ASIC	-	Application Specific Integration Circuit
ASIP	-	Application Specific Information Processing
ATSC	-	Advanced Television Systems Committee
DAC	-	Digital to Analog Converter
DNS	-	Directional Non-maximum Suppression
DSP	-	Digital Signal Processor
FPGA	-	Field Programmable Gate Array
HD	-	High Definition
IP	-	Internet Protocol
LED	-	Light Emitting Diode
NTSC	-	National Television Standards Committee
PAL	-	Phase Alternating Line
PLD	-	Programmable Logic Device
RTL	-	Register Transfer Level
SDRAM	-	Synchronous Dynamic Random Access Memory
SOPC	-	System on a Programmable Chip
SRAM	-	Static Random Access Memory
USB	-	Universal Serial Bus
VTR	-	Video Tape Recorder

LIST OF SYMBOLS

Tl	–	Low Threshold Value
Th	–	High Threshold Value
Hz	–	Hertz
$MBps$	–	Megabytes Per Second
$Mbps$	–	Megabits Per Second
M	–	Mega
σ	–	Standard Derivative

CHAPTER 1

INTRODUCTION

1.1 Background of the Study

A new generation of cameras that have been appeared since the late 1990s is called smart cameras. A smart camera is not just a camera that take pictures but it can perform tasks and do some processing depending on situations. Motion detection, object measurement, read vehicle number plates and even recognizing human behavior are some examples of smart camera processing [1].

Based on the quality of a camera, the size of the frames is different. Large frame requires high bandwidth communication. The size of an 24bit uncompressed digital 720×480 NTSC (National Television Standards Committee) frame is about 8.3Mega bit per frame. For 30 frames per second, it is around 250Mega bit per second. Furthermore, it may not necessary to send all data of frame as some especial data of the image are sufficient. The gradient of an image can be used to extract important detail of the image [2]. If the 24bit image is binarized, a speed of 1.3MBps or 10.4Mbps for wireless image transfer is sufficient. Canny algorithm is one of the optimum methods to compute the gradient of an image [3].

1.2 Statement of the Problem

Canny algorithm requires very high computation power because it needs a large amount of computation for different steps such as smoothing, thinning and thresholding. These steps contain many functions that are using multiplications, divisions and even more complex operations such as $Arctan$ which are very time consuming.

Achieving this level of processing power using programmable DSP (Digital Signal Processor) requires multiple processors. In order to efficiently use the hardware resources and increase the speed, hardware features like pipelining must be employed. A single FPGA (Field Programmable Gate Array) with an embedded soft processor can deliver the requisite level of computing power more cost-effectively, while simplifying board complexity [4].

1.3 Objective of the Study

The main objective of this project is to propose a software and hardware co-design architecture of Canny edge detection algorithm using FPGA for a fast image analysis. To achieve this aim, it is necessary to know the mathematical properties of edge detection. Therefore, the first objective of the project is to analyze the mathematical properties of Canny edge detection. The second objective is to map the mathematical operations as a hardware architecture model. Then several hardware/software Canny architectures are analyzed to evaluate each architecture trade-off. Finally execution time for every step in hardware and software compare together to determine which step should be done in software and which one is faster if implemented in hardware.

1.4 Scope of Work

The Modified Canny algorithm is benchmarked with a Matlab based implementation. The algorithm is simulated in Altera Quartus II and implemented in hardware to make a binary image of input image that contains just desired features, and has a less size in compare with original input image. The hardware architecture is based on RTL (Register Transfer Level) design methodology on Altera FPGA and input the image is a 24bit bitmap image.

1.5 Methodology

Figure 1.1 illustrates the steps of high-level implementing algorithms in hardware. At first step, RGB image is converted to grayscale to be ready for computing

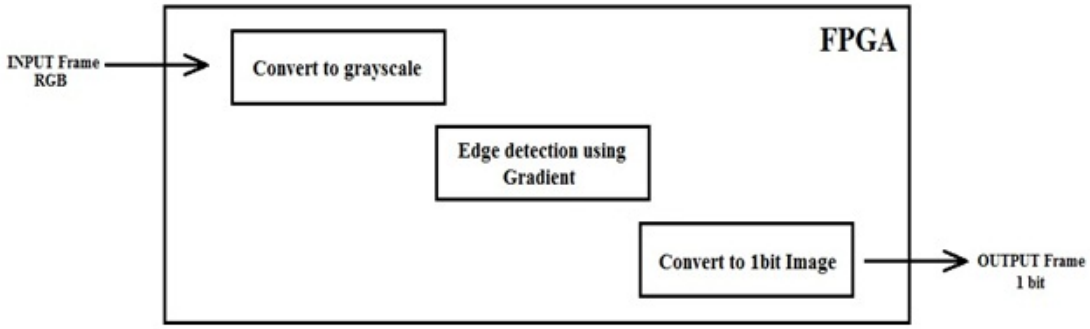

Figure 1.1: Steps of implementing algorithms on hardware.

gradient, in the next step the magnitude of first order derivative in x and y directions are computed using the Canny algorithm and are added together to yields gradient of the image. At last step the image is compressed as much as possible to get minimum size for a fast image transfer.

Canny algorithm step is written in Matlab (without modification). Some parts of Canny algorithm i.e. $Arctan$, are modified and compared with results from step 1. If the difference between results is negligible then modification is applied. The modified algorithm is written in NiossII Processor by Altera for various models of its FPGAs. execution time of every step is measured. Each processing step is separately analyzed as hardware structure. If the execution time is faster in software, the step is written in software else, it is implemented in hardware.

1.6 Report Outline

This rest of the report is organized as follows. Chapter 2 introduces a brief about smart cameras and Canny algorithm and it's complexity, DE2 Altera board, PAL (Phase Alternating Line) and NTSC standards and bitmap image. Chapter 3 describes how Canny algorithm modified to suit hardware implementation. Chapter 4 talks about the designed hardware architecture. Chapter 5 illustrates and compares the theoretical and experimental results. finally Chapter 6 summarizes this project and proposes directions for future work.

REFERENCES

1. Shi, Y. and Real, F. D. *Smart Cameras: Fundamentals and Classification*, Springer US, chap. 2. 2009, 19–34.

2. Chen, G. H., Yang, C. L. and Xie, S. L. Gradient-Based Structural Similarity for Image Quality Assessment. *2006 IEEE International Conference on Image Processing*. 2006. 2929–2932.

3. Canny, J. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986. 8(6): 679–698.

4. Neoh, H. S. and Hazanchuk, A. Adaptive Edge Detection for Real-Time Video Processing using FPGAs. *Global Signal Processing Expo and Conference (GSPx 2004)*. 2004.

5. Real, F. D. and Berry, F. *Smart Cameras: Technologies and Applications*, Springer US, chap. 3. 2009, 35–50.

6. Shi, Y. and Lichman, S. Smart Cameras: A Review. *Proceedings of 2005 Asia-Pacific Workshop on Visual Information Processing*. 2005. 95–100.

7. Pflugfelder, R. and Micusik, B. *Self-Calibrating Cameras in Video Surveillance*, Springer US, chap. 9. 2009, 161–180.

8. Cavallaro, A. *Change Detection for Object Segmentation*, Springer US, chap. 10. 2009, 181–198.

9. Dominguez, G. F., Beleznai, C., Lizenberger, M. and Delbruck, T. *Object Tracking on Embedded Hardware*, Springer US, chap. 11. 2009, 199–223.

10. Wang, X. and Jin, J. Q. An Edge Detection Algorithm Based on Improved CANNY Operator. *Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007)*. 2007. 623–628.

11. He, W. and Yuan, K. An Improved Canny Edge Detector and its Realization on FPGA. 2008. 6561–6564.

12. Wang, B. and Fan, S. An Improved CANNY Edge Detection Algorithm. *Second International Workshop on Computer Science and Engineering (WCSE 2009)*. 2009. 497–500.

13. Zhao, H., Qin, G. and Wang, X. Improvement of canny algorithm based on pavement edge detection. *3rd International Congress on Image and Signal Processing*. 2010. 964–967.

14. Rao, D. V. and Venkatesan, M. An efficient reconfigurable architecture and implementation of edge detection algorithm using Handle-C. *International Conference on Information Technology: Coding and Computing (ITCC 2004)*. 2004. 843–847.

15. Gonzalez, R. C. and Woods, R. E. *Digital Image Processing*. 2nd ed. Prentice Hall. 2002.

16. DE2-70 DE2 Development and Education Board UserManual, 2007.

17. DE2 Development and Education Board UserManual, 2007. URL `www.altera.com`.

18. DE2 Development and Education Board UserManual, 2006. URL `www.altera.com`.

19. Keith, J. *NTSC, PAL, and SECAM Overview*, Newnes, chap. 8. 2007, 257 – 387.

20. Miano, J. *Compressed Image File Formats : JPEG, PNG, GIF, XBM, BMP*. Longman Pub Group. 1999.

21. Jack, K. *YCbCr to RGB Considerations*. Technical report. Intersil. 1997.

22. Rao, D. V., Pati, S., Babu, N. A. and Muthukumar, V. Implementation and Evaluation of Image Processing Algorithms on Reconfigurable Architecture using C-based Hardware Descriptive Languages. *International Journal of Theoretical and Applied Computer Sciences*, 2006. 1(1): 9–34.

23. Nios II Software Developer's HandBook, 2011. URL `www.altera.com`.